Synergistic Office Solutions, Inc.

Topic: **Enhancing Performance of a Large Database**
Document ID: **#435**
Product: **SOS Office Manager for Windows**
Date: **August 31, 2010**
Author: **Seth Krieger**

Run the Database on a Dedicated System

For best performance, the database should be hosted on a system **dedicated to that one function**. If the system running the SOS database is also used to host other applications (for example: Exchange, Microsoft SQL Server, or Terminal Services), the performance of the database, as well as the other applications, will suffer during times of intensive activity.

Note that you can host the database on a "workstation" level operating system such as Windows XP Pro, Vista Business, or Windows 7 Pro. Microsoft restricts these environments to share resources with no more than 10 concurrent users, but the SOS database does not require any network shares, not does it require those accessing the database to log into the system that is hosting the database. *In fact, for security reasons SOS strongly discourages any network share that would expose the files in the server's SOS directory to network users (except for appropriate administrator-level staff).* All communication between the SOS users and the database is conducted through IP messaging, which is not restricted by Windows.

Use Appropriate Server Hardware

The larger your database, and the more simultaneous users, the more powerful your server must be to provide good performance. Specifically, there are three main considerations: RAM, drive sub-system, and processor(s). If you want the best possible performance, and the budget will allow, then run the database on a dedicated multi-processor system, with the fastest type of RAM, in a quantity that matches or exceeds the size of your database, and with a set of two or three of the fastest hard drives, or disk array that you can find.

System RAM

With a large database, the bottleneck for data access will always be the time it takes to read data from the drive system. By equipping your database server with plenty of RAM, the database engine will be able to cache the hard drive data in RAM, making data access many times faster than it could ever be, even with the fastest possible mechanical disk drives. The best possible situation would be one in which there is so much available RAM that a copy of virtually the entire database will end up in the cache. With the price of memory constantly falling, adding RAM is the easiest and least expensive way to significantly increase the performance of your database.

When purchasing a new computer to use as a database server, be sure to consider its maximum RAM capacity and the speed of the RAM it can use. Configure the server system with the expectation that you will want to increase the amount of RAM at some point. You should therefore purchase your RAM modules in sizes that will leave memory slots unused. For example, if you have four memory slots and

want to start with 1 GB of RAM, purchase two 512 MB modules so that two slots are left unused. If you were to purchase four 256 MB modules instead, you would have to discard all four if you later found you needed 2 GB of RAM.

Database Cache Configuration

In addition to the normal hard drive caching done by Windows, the database engine also creates and manages its own cache to optimize the performance of the database. The amount of RAM used as database cache is normally determined by settings in the SERVER.PRM file in your SOS program directory (by default, C:\SOS). SERVER.PRM is a small text file that can be opened and edited using Notepad or any other text editor.

Inspecting this file, you may see a line indicating a cache size. For example, -c 100m which would indicate an initial database cache size of 100 megabytes. Change the "100" to the desired number of megabytes. Alternatively, you can force the cache to a fixed size that will not change using -ca. For example, to configure an unchanging cache size of 150 MB, you would use the parameter: -ca 0 -c 150m. (-ca 0 disables dynamic cache resizing and -c 150m sets the desired cache size). To allow the cache size to change as needed but never below 100 MB, use the -cl parameter: -cl 100m. To keep the cache size from ever exceeding 250 MB, use -ch, such as: -ch 250m. You can use both -cl and -ch to define a range within which the size can automatically adjust, such as:

-cl 100m -ch 250m

For best performance, your database cache should be set to a value of at least half the size of your combined DB files (in \SOS\DATA). The cache will never grow beyond 256 MB unless you explicitly specify a larger -ch value. In 32-bit versions of Windows, the maximum size for the database cache is 1800 MB (-ch 1800m) unless you apply some special techniques that allow larger caches via Address Windowing Extensions (AWE). Contact SOS for more information.

[SOS 2010 and later] If you are running the database using the default SOS installation on a 64-bit version of Windows, and have plenty of RAM installed, you can go as high as 3.8 GB (-ch 3800). If your SOS server is a 64 bit Windows server, however, SOS recommends that you switch to the included 64 bit database engine, in which case you can set the cache high limit pretty much as high as you like, though the actual maximum size will be reset based on the amount of physical RAM in your system and your other system requirements. The next section explains how to reconfigure your system to use the 64-bit database engine. Remember that you cannot run the 64-bit engine unless you are running the database on 64-bit Windows. To check, right-click "Computer" or "My Computer" on the desktop or Start menu, then select Properties. If you do not see a notation on that screen that you are running a 64-bit version, you can assume that your copy of Windows is 32-bit.

[SOS 2010 and later] Switching to the 64-bit Database Engine

Again, as explained in the previous paragraph, you cannot run the 64-bit database engine unless you are running the database on a 64-bit version of Windows.

You can run the database as a foreground application or as a Windows Service. If the former, all you have to do is to change the shortcut(s) that you use to start the database to reference \SOS\BIN64\DBSRV11 instead of \SOS\BIN32\DBSRV11 (for network servers) or \SOS\BIN64\DBENG11 instead of \SOS\BIN32\DBENG11 (for standalone installations). To avoid confusion in the future, make sure that you make the change in all the shortcuts and command files that you use to start the engine. Change the command in:

- The Start Menu shortcut in **Start** > **SOS Applications**.
- Any shortcuts on the desktop used to start the database.
- Any additional shortcuts "pinned" to the Start menu or Task Bar.
- The command line on the ODBC data source named SOSDATA.
  1. Go to **Start** > **SOS Applications** > **ODBC** > **ODBC Administrator**.
  2. Click the **System DSN** tab.
  3. Double-click SOSDATA.
  4. Select the **Database** tab.
  5. In *Start line*, change "BIN32" to "BIN64".
- If you have and CMD or BAT files you use to start the database, make the change there as well.

If running as a service, you will have to delete the existing service and re-create it, using the appropriate command line, with the change from BIN32 to BIN64. See document 430 here: http://www.sosoft.com/fod/doc430-startingdbasservice.pdf

Drives

The database consists of two parts: the *database files* and the *transaction log*. By default, both are located in the DATA folder beneath SOS on the installation drive. This configuration is convenient, but not optimal for performance. Ideally there should be two separate drives. The program files and transaction log (SOSDATA.LOG) would be installed on the first drive, and the database files (the 11 files with the DB extension) would be installed on the second drive. In addition, in order to minimize file fragmentation, the database files would be placed in a partition that contains no other files.

The reason to place the log and the database files on different disks is because normal database operations involve reads and writes to both the transaction log file and the database files. When these files are on different disks, database accesses do not have to wait for log updates to be completed.

It is important to note that separate partitions and drive letters on the same physical hard disk will *not* provide the same benefits, and might even result in worse performance. Significant performance benefits will be realized only if the database is moved to its own physical drive, separate from the transaction log.

You cannot accomplish the above configuration by simply moving the files manually. The separation of the LOG file from the database files requires that the name of the log file be reset using the DBLOG utility. Failure to do so will result in the creation of a new log file in the same location as the database files when you restart the database. (See the procedure in the box below.)

Moving the Transaction Log File: Use the *DBLOG* utility, located in the \SOS\ASA\WIN32 folder, to move the transaction log to a different drive by setting a new transaction log name. Simply copying the file from one drive to another will not work because the location of the log is stored in the database.

1. Make sure that the database is not running.

2. *Move the* 11 database files from the \SOS\DATA folder to the new location. For example, we will assume that the second drive's designation is E: and that you have created a folder on E: named SOSDB. Further, we will assume that the original SOS installation is in the default location, C:\SOS. When you are finished, check to be sure that the DB files are no longer present in the default location to minimize confusion in the future. SOS suggests that you create a text file called DB_LOCATION.TXT in the \SOS\DATA folder in which you document the location of the production database. The \SOS\DATA folder will then contain just two files: SOSDATA.LOG and DB_LOCATION.TXT.
3. Open a command window (Start > Run, CMD <enter>) and change to the C:\SOS\ASA\WIN32 directory. At the C:\SOS\ASA\WIN32 prompt, type, in lower case:
   `dblog -t` *<new log name>* *<database name>*

   The *new log name* should include the drive, path, and filename of the log file, for example: **`c:\sos\data\sosdata.log`**

   The *database name* should include the drive, path, and filename of the main database file, for example, **`e:\sosdb\sosdata.db`**.
4. Modify your database startup command in the menu item, desktop shortcut, and/or Windows service configuration to reflect the new location of the DB files.
5. ***CRITICAL***: Modify your backup script or configuration to include both the SOSDATA.LOG file in C:\SOS\DATA, and the 11 DB files in E:\SOSDB. **Test it to be sure you are picking up these files, then test it again to be ABSOLUTELY sure!**

How about RAID? Use of a performance-oriented RAID array, such as RAID-5, would be of benefit if just the database files were stored on the array, and the log file were left on another disk. Data is simply appended to the end of the log file, not accessed randomly, so there is little to be gained by locating it on the array, and it would still be more efficient to have the log on a separate disk.

Processor

All other things being equal, a faster processor will deliver the results in less time than a slower processor. Nevertheless, if you have adequate processing power, start with RAM and drive improvements before upgrading to a faster processor. In almost all cases, performance will be bound by the speed at which data can be delivered to the processor(s), not the speed at which the processor executes instructions. For this reason, RAM and drive system improvements are likely to be of more benefit than processor upgrades.

That being said, there are definite advantages to hosting the database on a system with multiple processors. When there are two or more processors in the server, more than one query or database request can be processed at the same time, minimizing log-jams at installations with many users. Your basic network license with SOS allows use on up to four processors on your server system. Systems with more than four processors require additional licensing to stay legal with Sybase, but the cost is very modest. Call SOS for details.

Maintenance

Because disk performance is so critical to the speed at which the database can process requests, anything you do to make disk access more efficient will result in better overall performance. Two maintenance procedures are significant in this regard: disk defragmentation and database rebuilds.

When you defragment your hard disk, you rearrange disk contents so that each file is located in contiguous sectors of the drive, minimizing the amount of arm movement needed to retrieve data from different parts of the database, and minimizing the overhead necessary to locate pieces of the file spread out on the drive. Doing so can make a very significant difference in performance. You can defragment your drive(s) using the utility included with Windows (right-click the drive from **My Computer**, then select **Properties**, then **Tools**), or using a commercial product such as DiskKeeper from Executive Software (http://www.executive.com).

Note: The database must be shut down while doing the defragmentation.

From time to time, a database rebuild reorganizes the data within each of the physical DB files, making the physical files as small and efficient as they can be. The rebuild also recreates all the database indices, improving data access speed. For complete instructions, see:

http://www.sosoft.com/fod/doc127-dbrebuild.pdf

If your database and log file are not located in the same drive and folder, you may have to reset the log file name (as described in step 3 above) after your rebuild is complete.